

STRUCTURED DOCUMENT EDITING DEVICE

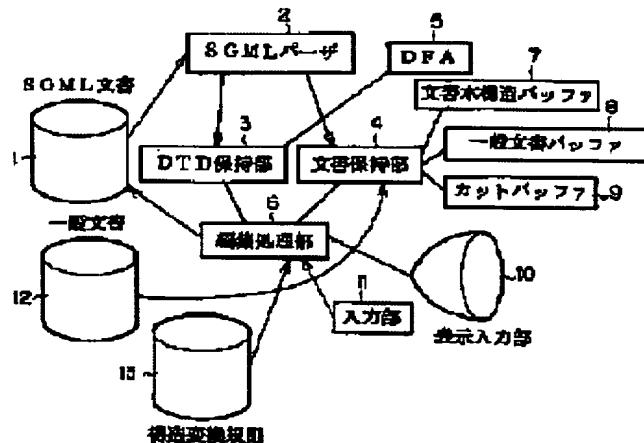
Patent number: JP8016576
Publication date: 1996-01-19
Inventor: SENDA SHIGEYA
Applicant: RICOH KK
Classification:
 - international: G06F17/24; G06F17/24; (IPC1-7): G06F17/24
 - european:
Application number: JP19940150868 19940701
Priority number(s): JP19940150868 19940701

[Report a data error here](#)

Abstract of JP8016576

PURPOSE: To efficiently perform an editing work by eliminating a substitution work by confirming the substitution of a real reference by a user afterward by substituting the real reference when a paste is performed.

CONSTITUTION: An editor receives an SGML document 1 as an input and stores an edition result as an output by an SGML document form. In the case of a new document, a DTD is designated by a user and the only DTD is read. When the editor reads the SGML document/DTD, the input is analyzed by an SGML parser 2 at first. The SGML parser 2 prepares a finite automata (DFA) 5 corresponding to the DTD as an analysis result and stores the automata in a DTD holding part 3. The DFA 5 expresses a list where the lower rank node of each tag is possible. When the character string matching with the real declaration in an insertion destination document exists in a text area when the text area is inserted into a structured document, the character string is substituted for a real reference.



Data supplied from the **esp@cenet** database - Worldwide

a)

(19)日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平8-16576

(43)公開日 平成8年(1996)1月19日

(51) Int.Cl.⁶
G 0 6 F 17/24識別記号
9288-5L

府内整理番号

F I

技術表示箇所

G 0 6 F 15/ 20

5 5 4 H

審査請求 未請求 請求項の数 6 O.L. (全 9 頁)

(21)出願番号 特願平6-150868

(22)出願日 平成6年(1994)7月1日

(71)出願人 000006747

株式会社リコー

東京都大田区中馬込1丁目3番6号

(72)発明者 千田 滋也

東京都大田区中馬込1丁目3番6号 株式
会社リコー内

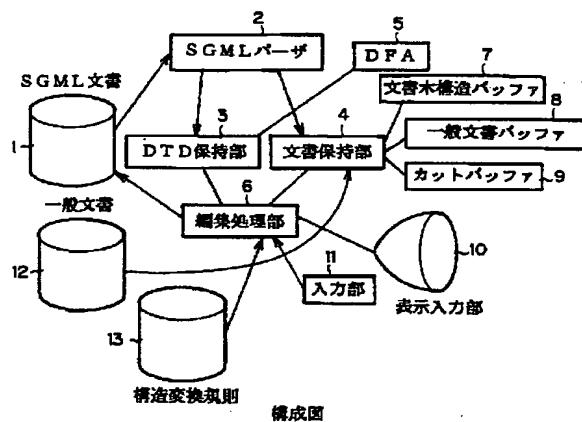
(74)代理人 弁理士 高野 明近

(54)【発明の名称】構造化文書編集装置

(57)【要約】

【目的】ペーストする際に実体参照に置き換えることにより、ユーザがあとから確認して置き換える作業がいらなくなるようにし、編集作業の効率化を図る。

【構成】エディタはSGML文書1を入力として受取り、編集結果を出力としてSGML文書形式で格納する。新規文書の場合、使用者にDTDを指定してもらい、DTDのみを読み込む。エディタがSGML文書・DTDを読み込む場合、まず入力をSGMLパーザ2で解析する。該SGMLパーザ2は解析結果としてDTDに対応して有限オートマトン(DFA)5を作成し、DTD保持部3に格納する。DFA5は各タグの下位のノードの可能な並びを表現する。構造化文書中にテキスト領域を挿入する際に、挿入先文書にある実体宣言に一致する文字列がテキスト領域にある場合、該文字列を実体参照に置き換える。



1

【特許請求の範囲】

【請求項1】 構造化文書を保持する文書保持部と、該構造化文書の編集処理を行う編集処理部と、文書内容の一部を替えるバッファとを具備し、構造化文書中にテキスト領域を挿入する際に、挿入先文書にある実体宣言に一致する文字列がテキスト領域にある場合、該文字列を実体参照に置き換えることを特徴とする構造化文書編集装置。

【請求項2】 構造化文書を保持する文書保持部と、該構造化文書の編集処理を行う編集処理部と、文書内容の一部を替えるバッファとを具備し、構造化文書中にテキスト領域を挿入する際に、挿入元文書にある実体宣言が未定義である場合、該実体宣言を挿入先文書に複写することを特徴とする構造化文書編集装置。

【請求項3】 構造化文書を保持する文書保持部と、構造に関する情報を保持するDTD保持部と、編集過程の情報を表示する表示入力部と、構造化文書の編集処理を行う編集処理部と、文書内容の一部を替えるバッファとを具備し、構造化文書中の部分構造を対象として削除、複写、挿入を行うことを特徴とする構造化文書編集装置。

【請求項4】 前記部分構造を挿入する際、挿入先文書の文書構造に適合するノード位置を表示することを特徴とする請求項3記載の構造化文書編集装置。

【請求項5】 前記部分構造を挿入した際、挿入した部分構造中の文書構造に適合しないノードの情報を表示することを特徴とする請求項3記載の構造化文書編集装置。

【請求項6】 前記部分構造を挿入する際、挿入元と挿入先の構造間の変換規則により変換を行った後に挿入することを特徴とする請求項3記載の構造化文書編集装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、構造化文書編集装置に關し、より詳細には、ペースト(paste)する際に実体参照に置き換えることにより、ユーザがあとから確認して置き換える作業がいらなくなるようにし、編集作業の効率化を図るようにした構造化文書編集装置に関する。

【0002】

【從来の技術】 従来の構造文書編集装置について記載した公知文献としては、例えば、特開平2-247772号公報がある。この公報のものは、ユーザがタブ表示を必要とするかどうかに係わりなく、文書階層の要素論理構造を操作することを可能にし、標準一般化マークアップ言語(SGML)によって定義されるようなテキストやグラフィック・イメージ又は他のデータを含む階層的文書構造の要素をマークするものである。すなわち、構造文書編集において、カーソルのある位置のエレメントを含む構造の階層を表示するものである。

2

【0003】 SGML規格は、構造化文書の交換を標準化するための規格である。文書の構造は文書型定義(DTD, Document Type Definition)によって定義される。DTDで規定された文書構造にしたがって文書を作成する場合、プレーンテキストとしてタグを内容と同時にに入力し、バッチ処理によってSGML文書を処理するか、専用のエディット機能を持つ構造化文書エディタを利用する必要がある。

【0004】

【発明が解決しようとする課題】 文書を標準化し、多量の文書を効率良く管理、運用するために構造化文書(SGML)を用いることが考えられる。しかし、現状ではSGML文書ではなく他の記述形式で書かれた文書、例えば、通常のワードプロセッサなどで書かれた文書などがほとんどで、それらをSGML化する作業が必要である。これらの作業のために文書の変換処理を行う機構が考えられる。変換機構は、文書構造がはっきり分かっている場合はバッチ的に処理可能である場合もあるが、そうでない場合、人間の介入が必要になって来る。従来のエディタでは、従来文書からの変換に関してはほとんど考慮されておらず、単にテキストのcut&pasteなどが用意されているだけであった。

【0005】 本発明は、このような実情に鑑みてなされたもので、ペーストする際に実体参照に置き換えることにより、ユーザがあとから確認して置き換える作業がいらなくなるようにし、編集作業の効率化を図ること、また、ペーストする際に未定義の実体宣言を複写先にコピーすることにより、ユーザがあとから新たに宣言する必要をなくし、編集作業の効率化を図ること、さらに、部分構造を構造文書間で削除、複写、挿入することで、同様の内容を新たに作ることなく文書を作成でき、編集作業の効率化を図るようにした構造化文書編集装置を提供することを目的としている。

【0006】

【課題を解決するための手段】 本発明は、上記課題を解決するために、(1)構造化文書を保持する文書保持部と、該構造化文書の編集処理を行う編集処理部と、文書内容の一部を替えるバッファとを具備し、構造化文書中にテキスト領域を挿入する際に、挿入先文書にある実体宣言に一致する文字列がテキスト領域にある場合、該文字列を実体参照に置き換えること、或いは、(2)構造化文書を保持する文書保持部と、該構造化文書の編集処理を行う編集処理部と、文書内容の一部を替えるバッファとを具備し、構造化文書中にテキスト領域を挿入する際に、挿入元文書にある実体宣言が未定義である場合、該実体宣言を挿入先文書に複写すること、或いは、(3)構造化文書を保持する文書保持部と、構造に関する情報を保持するDTD保持部と、編集過程の情報を表示する表示入力部と、構造化文書の編集処理を行う編集処理部と、文書内容の一部を替えるバッファとを具備

し、構造化文書中の部分構造を対象として削除、複写、挿入を行うこと、更には、(4)前記(3)において、前記部分構造を挿入する際、挿入先文書の文書構造に適合するノード位置を表示すること、更には、(5)前記(3)において、前記部分構造を挿入した際、挿入した部分構造中の文書構造に適合しないノードの情報を表示すること、更には、(6)前記(3)において、前記部分構造を挿入する際、挿入元と挿入先の構造間の変換規則により変換を行った後に挿入すること、或いは、(7)一般文書と構造化文書を保持する文書保持部と、構造に関する情報を保持するDTD保持部と、編集過程の情報を表示する表示入力部と、文書の編集処理を行う編集処理部とを具備し、一般文書の内容を削除、複写し、その内容を挿入する際に構造文書の注目点の次に来るデータ内容までの要素列を内容モデルから作成し、該内容モデルを使用者にリストとして示して選択させ、その要素列を作成して一般文書の内容をデータ内容として挿入することを特徴としたものである。

【0007】

【作用】前記構成を有する本発明の構造化文書編集装置は、構造化文書を保持する文書保持部と、該構造化文書の編集処理を行う編集処理部と、文書内容の一部を替えるバッファとを具備し、(1)構造化文書中にテキスト領域を挿入する際に、挿入先文書にある実体宣言に一致する文字列がテキスト領域にある場合、該文字列を実体参照に置き換えるので、ペーストする際に実体参照に置き換えることにより、ユーザがあとから確認して置き換える作業がいらなくなるため、編集作業が効率的に行える。(2)構造化文書中にテキスト領域を挿入する際に、挿入元文書にある実体宣言が未定義である場合、該実体宣言を挿入先文書に複写するので、ペーストする際に未定義の実体宣言を複写先にコピーすることにより、ユーザがあとから新たに宣言する必要がなく、編集作業が効率的に行える。

【0008】また、構造化文書を保持する文書保持部と、構造に関する情報を保持するDTD保持部と、編集過程の情報を表示する表示入力部と、構造化文書の編集処理を行う編集処理部と、文書内容の一部を替えるバッファとを具備し、(3)構造化文書中の部分構造を対象として削除、複写、挿入を行うので、部分構造を構造文書間で削除、複写、挿入することで同様の内容を新たに作ることなく文書を作成でき、編集作業が効率的に行える。(4)前記部分構造を挿入する際、挿入先文書の文書構造に適合するノード位置を表示するので、部分構造を構造文書間で削除、複写、挿入する際に内容モデルを壊すことなく挿入できる位置を見付けることができ、編集作業が効率的に行える。(5)前記部分構造を挿入した際、挿入した部分構造中の文書構造に適合しないノードの情報を表示するので、部分構造を構造文書間で削除、複写、挿入した際に内容モデルに違反する内容を見

付けることができ、編集作業が効率的に行える。(6)前記部分構造を挿入する際、挿入元と挿入先の構造間の変換規則により変換を行った後に挿入するので、部分構造を構造文書間で削除、複写、挿入する際に、部分構造を複写先に適合するよう変換することができ、編集作業が効率的に行える。

【0009】さらに、一般文書と構造化文書を保持する文書保持部と、構造に関する情報を保持するDTD保持部と、編集過程の情報を表示する表示入力部と、文書の編集処理を行う編集処理部とを具備し、(7)一般文書の内容を削除、複写し、その内容を挿入する際に構造文書の注目点の次に来るデータ内容までの要素列を内容モデルから作成し、該内容モデルを使用者にリストとして示して選択させ、その要素列を作成して一般文書の内容をデータ内容として挿入するので、一般的な文書から構造文書への変換の際、構造を簡単に作成することができ、編集作業が効率的に行える。

【0010】

【実施例】実施例について、図面を参照して以下に説明する。図1は、本発明による構造化文書編集方式の一実施例を説明するための構成図で、図中、1はSGML文書、2はSGMLバーザ、3はDTD保持部、4は文書保持部、5はDFA(有限オートマトン)、6は編集処理部、7は文書木構造バッファ、8は一般文書バッファ、9はカットバッファ、10は表示入力部、11は入力部、12は一般文書、13は構造変換規則である。

【0011】エディタの入力はSGML文書であり、SGML文書1内にはDTDの指定またはDTDそのものが記述されている。エディタはSGML文書を入力として受取り、編集結果を出力としてSGML文書形式で格納する。新規文書の場合、使用者にDTDを指定してもらい、DTDのみを読み込む。エディタがSGML文書・DTDを読み込む場合、まず入力をSGMLバーザ2で解析する。該SGMLバーザ2は解析結果としてDTDに対応して有限オートマトン(DFA)5を作成し、DTD保持部3に格納する。DFA5は各タグの下位のノードの可能な並びを表現する。

【0012】図2は、1つのタグ“kaigi”についてのDFAを示す図である。DTDで定義される各タグ(エレメント)毎にDFA5が作成され、DTD保持部3にその他のDTDの情報(実体宣言、DTD名 etc...)とともに格納される。後の処理では、それを利用してSGML文書1の解析を続ける。SGML文書自体の解析の結果として内部に、図3に示すような文書構造に対応する木構造を作成する。作成した木構造は元になったDTDと関連づけて文書保持部4に格納される。木構造の各ノードには種別毎に情報が保持されている。

タグ：タグ名、属性

データ：文字列データ

processing instruction：文字列データ

【0013】図4は、図2、図3に対応するSGML文書のDTDの一部を示す図である。また、図5は、図4のDTDを使用した文書の例の一部を示す図である。また、図6は、表示部における画面例を示す図で、この画面では一般文書のウインドウと構造化文書のウインドウが示されている。図1において、カットバッファはcutまたはcopyの操作によって一般文書、文書木構造中のテキストまたは文書木構造の部分構造を保持する。入力部(キーボード、マウス等)によって図6の画面上でテキスト、構造を指定し、コマンドを実行することでカットバッファにその内容を入力する。

【0014】図7は、カットバッファに内容をpaste(ペースト)する際の動作を説明するためのフローチャートである。以下、各ステップ(step)に従って順に説明する。まず、テキストのみかどうかを判断し(step1)、テキストのみであれば、次に文字列挿入位置を得(step2)、テキスト置換処理を行う(step3)。次に、挿入位置に文字列を挿入する(step4)。前記step1において、テキストのみでなければ、構造挿入位置を得(step5)、ノードのテキスト毎にテキスト置換処理を行う(step6)。次に、挿入位置に構造を挿入する(step7)。

【0015】この動作は入力部からの指示によって起動される。カットバッファに取り込まれた内容は取り込まれた元の文書の情報と内容を保持している。内容はテキストのみか構造部分木(請求項3)である。そのどちらかによって処理が分かれる。テキストのみの場合、挿入位置を得てテキスト置換処理を行い、その結果を挿入する。部分構造の場合、挿入位置を得て部分構造のテキストを含むノードに対してテキスト置換処理を行う。従来はテキストのみのcut&pasteが行われているのみであったが、請求項3においては、単に文字列のcut&pasteだけでなく、構造単位でcut&pasteでき、便利である。

【0016】図8は、テキスト置換処理を説明するためのフローチャートである。以下、各ステップ(step)に従って順に説明する。まず、テキスト中にある実体参照があるとき、それが複写先にない場合、複写元の実体宣言を複写先にコピーする(step11)。次に、複写先で定義されている実体宣言と一致する文字列をその実体参照に置き換える(step12)。すなわち、元テキスト中に実体参照がある場合で、その実体参照が複写先文書に存在しない場合、それを複写先に登録する(請求項2)。このように、請求項2においては、cutされたテキスト中に実体参照がある場合、元の文書では定義されているが、挿入された文書では未定義である可能性がある。こういった場合に、自動的に挿入先文書に実体定義を複写する機能が考えられる。

【0017】逆に挿入先文書にある実体宣言と一致する文字列がテキスト中にある場合、それを実体参照に置換する(請求項1)。このように、テキストのcut&paste

において、従来は単純に文字列をcut&pasteするだけであったが、その文字列中に複写先において実体宣言がなされている文字列が含まれている場合が考えられる。請求項1においては、これらの文字列を実体参照にpasteの際、置き換えることで挿入先での文書の利用がやりやすくなる。

【0018】図9は、図7における構造挿入位置を得るための処理を説明するためのフローチャートである。以下、各ステップ(step)に従って順に説明する。まず、一般構造変換処理を行い(step21)、挿入する構造木のトップノードをxとする(step22)。次に挿入先文書のノードを1つとり、yとし(step23)、yの子どもまたは兄弟としてxが来ることができるかDFAをサーチする(step24)。次に、xが来ることができるかどうかを判断し(step25)、来ることができなければ、前記step23へ行き、来ることができれば、ノードを兄弟の場合は青、子どもの場合は赤でハイライト表示する(step26)。

【0019】次に、全てのノードを調べたかどうかを判断し(step27)、調べてなければ、前記step23へ行き、調べてあれば、ユーザに挿入位置を選択してもらう(step28)。次に、選択位置がDTDに適合するかDFAを検索し(step29)、来ることができるかどうかを判断し(step30)、来ることができなければ、警告メッセージを表示する(step31)。次に、挿入された並びのコンテンツモデルをDFAでチェックする(step32)。挿入された構造のコンテンツモデルをDFAで再帰的にチェックし(step33)、両方のチェックでDFAが満足されなかった位置のノードを反転表示する(step34)。

【0020】テキストの場合は、単純にテキスト表示の位置を得るだけであるが、構造の場合、挿入する構造がDTDに適合することが重要である。まず、挿入できる位置をユーザに示すため、木構造をサーチし、挿入可能位置をDTDのDFAから求める。その位置をハイライト表示する(請求項4)。ノードに対し、挿入可能位置は兄弟、または子どもの位置の2通りが考えられる。

【0021】そこで、この実施例では兄弟の位置の場所ノードに対して青で、子ども位置の場所赤でハイライト表示する。ユーザは挿入先を選択するが、画面上では挿入位置のノードをクリックすることで行う。このときダイアログを表示し、兄弟として挿入するか、子どもとして挿入するかを選択する。このように、請求項4においては、構造のcut&pasteでは構造を挿入先に挿入するとき、DTDのコンテンツモデルに適合するノード位置を表示することで間違った挿入を防ぐことができる。

【0022】選択されたノードがDFAに適合しない場合、警告を表示する(請求項5)。また、適合する場合でも、挿入によってコンテンツモデルが崩れる場合がある。例えば、挿入位置に同じエレメントによる構造が存

在する場合である。この場合、同じエレメント名の構造が挿入によって並んでしまう。さらに、挿入された構造が挿入前の文書で正しいコンテンツモデルであっても挿入後、挿入先のDTDと適合するとは限らない。そこで、ならびに、挿入された構造をDFAでチェックし、適合しない場合、ノードを反転表示し、使用者に注意を促す。

【0023】このようなSGML文書間の部分構造のcut&pasteでは、DTDが異なっている場合、挿入された側の文書に同じelementがない場合や名前が同じでもコンテンツモデルが異なる場合が考えられる。請求項5においては、コンテンツモデルが適合しない場合、適合しない箇所を表示し、ユーザに変更を促すので便利である。

【0024】DTDの構造が元のDTDと挿入先で全く異なる場合、挿入した後、エレメント名の変更や構造の変更を多量に行わなければならない。全く異なる文書でも何らかの意図があってcut&pasteしているわけであるので、使用者にとっては規則的な対応関係があるはずである。そこで、決まっている対応関係がある場合、それを規則的に変更してやれば良い。異なるDTD文書の変換はDSSLで一般文書変換(GTLP, STTP)で行われている。これは文書全体を対象とした物であるが、ここではcut&pasteの部分構造木を対象とする。この処理が図9の最初の所で行われる(請求項6)。このように、請求項6においては、頻繁にcut&pasteする場合、自動的に変換できるようにルールを設定できればユーザの手間を省くことができる。

【0025】一般文書から構造化文書へ変換する場合、cut&pasteだけでは元の文書にない構造を先に変換先に作成しておいてからテキスト内容を挿入する必要があり、変換作業に手間がかかる。そこで、一般文書から構造化文書への変換の際にエディタにおいて変換支援モードを設ける。このモードにおいては、構造化文書中に挿入ポイントというものを設ける。挿入ポイントは構造木のノードに存在する。

【0026】図10は、変換支援モードペースト処理を説明するためのフローチャートである。以下、各ステップ(step)に従って順に説明する。まず、テキスト置換処理を行い(step41)、ノードバスリストxを空にする(step42)。次に、挿入ポイントのノードをyとし(step43)、yの子ども方向にバスを作成してxに登録する(step44)。yの兄弟方向にバスを作成してxに登録し(step45)、yの親をyとする(step46)。次に、yが存在するかどうかを判断し(step47)、yが存在しなければ、前記step45に戻り、yが存在すれば、バスリストを表示してバスを選択させる(step48)。バスからノードを作成し、カットバッファからテキストをデータノードに挿入する(step49)。

【0027】すなわち、構造支援モードでは図7で示したpaste処理と動作が異なる。図10ではまず図7と同様のテキスト置換処理(図8)が行われる。次に、挿入ポイントにおいて次のデータノードにたどり着くノードバス候補を求める。これは、DFAを挿入ポイントとその祖先のノードからたどることによって得られる。

【0028】例えば、図3においてkenmeiの所に挿入ポイントがあるとき、バスはkenmei->sakusei:name:

となる。ただし、“->”は弟を示し、“:”は子どもを示す。つまり、図6にある一般文書で“千田滋也”をcutし、挿入ポイントkenmeiで挿入すると、自動的にバスに従ってsakuseiとnameが作られ、図5に示すようにmynameの実体宣言があれば、実体参照(&myname;)に変換されてデータがnameの下に挿入される。

【0029】テキスト部分のcut&pasteだけでは他の文書形式から変換する際に不便である。入力文字列に対してユーザがSGMLの構造として指定する機能を付け加えることにより、より効率的に取り込むことが可能になる。つまり、取り込もうとしている部分に対する構造をユーザが指定する手段、および、システム側から指定を手助けするための候補を与えてやることにより、効率良く構造のない文書を構造を持つ文書構造に変換することができる。

【0030】

【発明の効果】以上の説明から明らかなように、本発明によると、以下のような効果がある。

(1) 請求項1に対応する効果：構造化文書を保持する文書保持部と、該構造化文書の編集処理を行う編集処理部と、文書内容の一部を蓄えるバッファとを具備し、構造化文書中にテキスト領域を挿入する際に、挿入先文書にある実体宣言に一致する文字列がテキスト領域にある場合、該文字列を実体参照に置き換えるので、ペーストする際に実体参照に置き換えることによりユーザがあとから確認して置き換える作業がいらなくなるため、編集作業が効率的に行える。

(2) 請求項2に対応する効果：構造化文書中にテキスト領域を挿入する際に、挿入元文書にある実体宣言が未定義である場合、該実体宣言を挿入先文書に複写するので、ペーストする際に未定義の実体宣言を複写先にコピーすることによりユーザがあとから新たに宣言する必要がなく、編集作業が効率的に行える。

(3) 請求項3に対応する効果：構造化文書を保持する文書保持部と、構造に関する情報を保持するDTD保持部と、編集過程の情報を表示する表示入力部と、構造化文書の編集処理を行う編集処理部と、文書内容の一部を蓄えるバッファとを具備し、構造化文書中の部分構造を対象として削除、複写、挿入を行うので、部分構造を構造文書間で削除、複写、挿入することで同様の内容を新たに作ることなく文書を作成でき、編集作業が効率的に

行える。

(4) 請求項4に対応する効果：前記部分構造を挿入する際、挿入先文書の文書構造に適合するノード位置を表示するので、部分構造を構造文書間で削除、複写、挿入する際に内容モデルを壊すことなく挿入できる位置を見付けることができ、編集作業が効率的に行える。

(5) 請求項5に対応する効果：前記部分構造を挿入した際、挿入した部分構造中の文書構造に適合しないノードの情報を表示するので、部分構造を構造文書間で削除、複写、挿入した際に内容モデルに違反する内容を見付けることができ、編集作業が効率的に行える。

(6) 請求項6に対応する効果：前記部分構造を挿入する際、挿入元と挿入先の構造間の変換規則により変換を行った後に挿入するので、部分構造を構造文書間で削除、複写、挿入する際に、部分構造を複写先に適合するよう変換することができ、編集作業が効率的に行える。

(7) 解決手段7に対応する効果：一般文書と構造化文書を保持する文書保持部と、構造に関する情報を保持するDTD保持部と、編集過程の情報を表示する表示入力部と、文書の編集処理を行う編集処理部とを具備し、一般文書の内容を削除、複写し、その内容を挿入する際に構造文書の注目点の次に来るデータ内容までの要素列を内容モデルから作成し、該内容モデルを使用者にリストとして示して選択させ、その要素列を作成して一般文書の内容をデータ内容として挿入するので、一般の文書から構造文書への変換の際、構造を簡単に作成することができ、編集作業が効率的に行える。

【図面の簡単な説明】

【図1】 本発明による構造化文書編集装置の一実施例を説明するための構成図である。

【図2】 本発明におけるDFAを示す図である。

【図3】 本発明における文書保持部の構造を示す図である。

【図4】 本発明におけるDTDの例を示す図である。

【図5】 本発明におけるSGML文書例を示す図である。

【図6】 本発明における表示部の画面例を示す図である。

【図7】 本発明におけるペースト処理を説明するためのフローチャートである。

【図8】 本発明におけるテキスト置換処理を説明するためのフローチャートである。

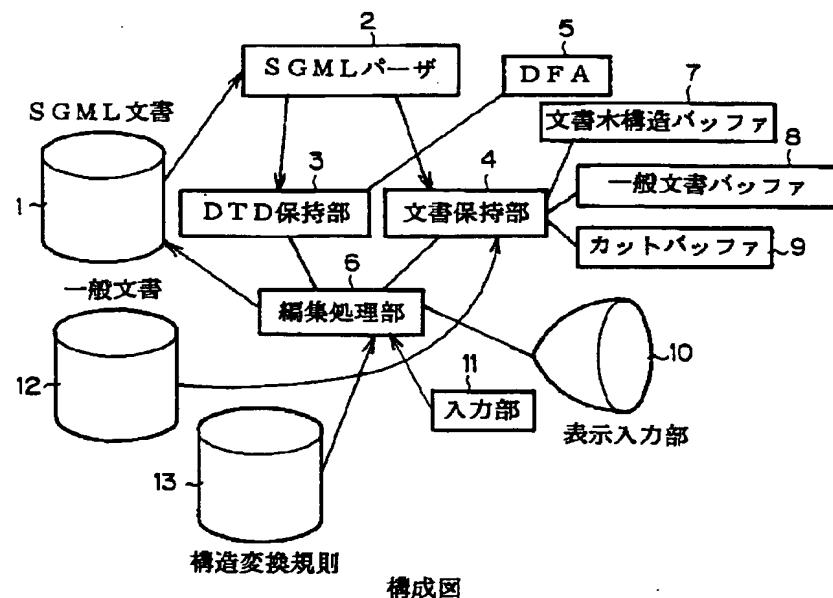
【図9】 本発明における構造挿入位置を保つ処理を説明するためのフローチャートである。

【図10】 本発明における変換支援モードペースト処理を説明するためのフローチャートである。

【符号の説明】

1…SGML文書、2…SGMLパーザ、3…DTD保持部、4…文書保持部、5…DFA（有限オートマトン）、6…編集処理部、7…文書木構造バッファ、8…一般文書バッファ、9…カットバッファ、10…表示入力部、11…入力部、12…一般文書、13…構造変換規則

【図1】



【図5】

```

<!DOCTYPE kaigi SYSTEM "DTD-kaigi"
[
  <!ENTITY &name "千田透也">
]
<kaigi>
<kennsei>HP ページアップについて</kennsei>
<akusei>
<name>&name;</name>
<lab>421</lab>
<phone type="short">3714</phone>
<email>sendai@ipr.rdc.ricoh.co.jp</email>

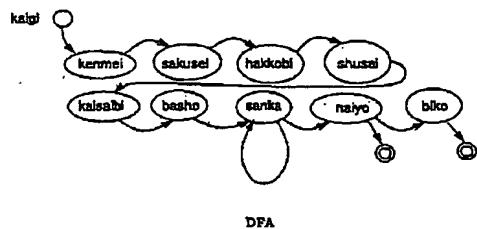
</akusei>
<nakohi>
<shugai>
<name>松田 透</name>
<lab>421</lab>
<phone type="short">3710</phone>

.....
</kaigi>

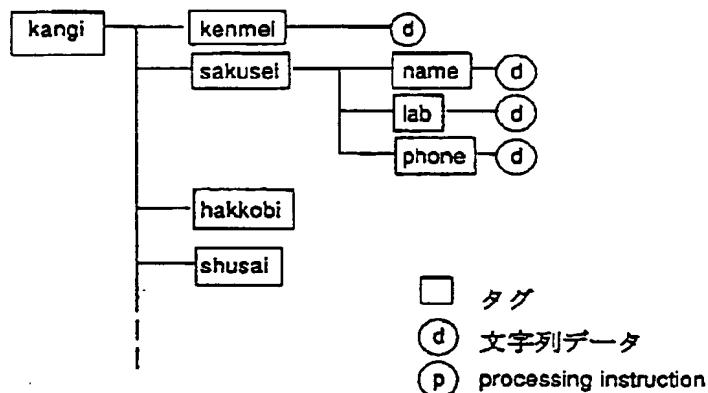
```

SGML文書

【図2】



【図3】



文書保持部の木構造

【図4】

```
<!--This is a document type definition for "会議開催通知" -->
<!ENTITY % person SYSTEM "(DTD)-prs" -include 人名所属 entity ->
<!ENTITY % bun SYSTEM "(DTD)-buninfo" -include 文 entity ->
<!ENTITY % nichiji "(tuki,hi,youbi,hajime,osari)">
% person;
% bun;

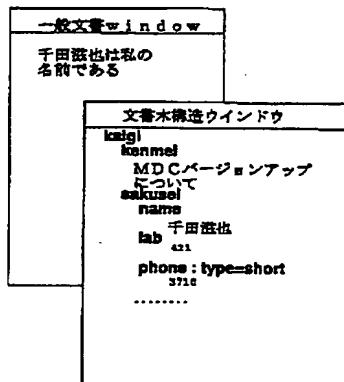
<ELEMENT kaijgi (kenmei,sakusei,hakkobi,shusai,kaisaihi,bisho,senka+,naiyo,biko)*
    >--作成者、実行日、件名、場所、参加者-->

<ELEMENT kenmei (%bun;)*>一件名-->
<ELEMENT sakusei (%buninfo;)*>一作成者-->
<ELEMENT hakkobi EMPTY>一実行日-->
<ELEMENT shusai (%buninfo;)*>一主催者-->
<ELEMENT kaisaihi (%ichidai;)*>一場所-->
<ELEMENT tuki (%CDATA)*>一月-->
<ELEMENT hi (%CDATA)*>一日-->
<ELEMENT youbi (%CDATA)*>一曜日-->
<ELEMENT hajime (%CDATA)*>一開始時刻-->
<ELEMENT osari (%CDATA)*>一終了時刻-->
<ELEMENT bisho (%CDATA)*>一場所-->
<ELEMENT senka (%buninfo;)*>一参加者-->
<ELEMENT naiyo (%agenda;)*>一内容-->
<ELEMENT agenda (%e,p;)*>一事項-->
<ELEMENT biko (%p;)*>一指考-->

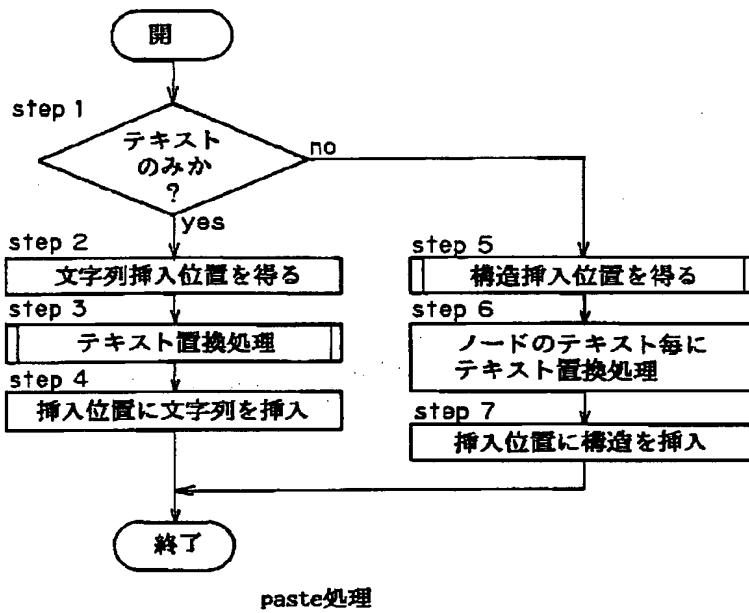
<!--Because this is DTD, entities need to be called using '%' but not 'a' that
     is used in instances. -->
```

DTDの例

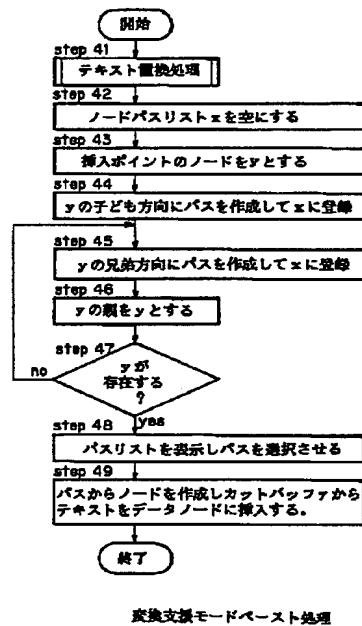
【図6】



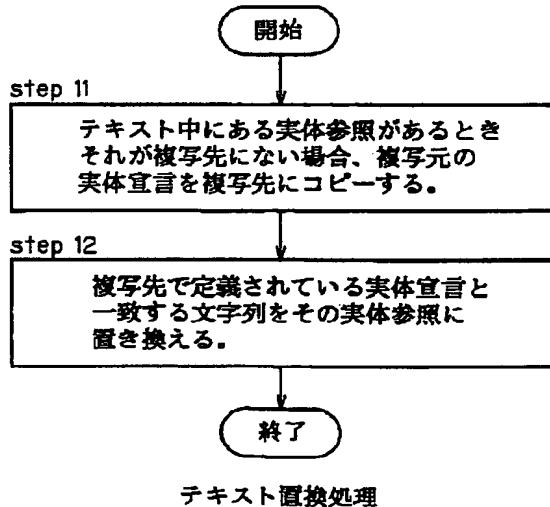
【図7】



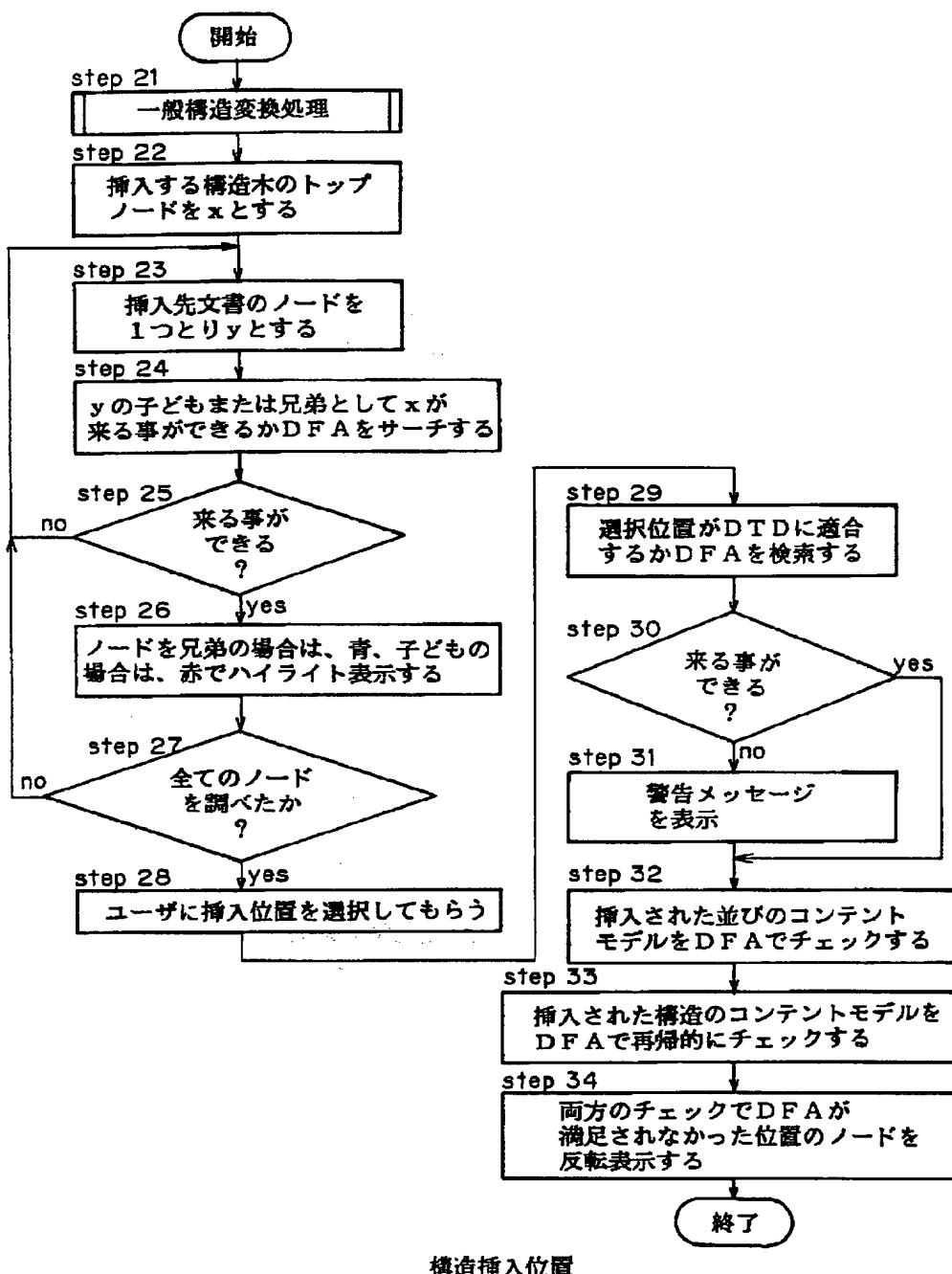
【図10】



【図8】



【図9】



構造挿入位置